(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) Publication number : **0 657 810 A1**

(12) EUROPEAN PATENT APPLICATION

(21) Application number : **94309029.0**

(51) Int. Cl.[6] : **G06F 9/46**

(22) Date of filing : **05.12.94**

(30) Priority : **06.12.93 US 161811**

(43) Date of publication of application :
14.06.95 Bulletin 95/24

(84) Designated Contracting States :
DE FR GB IT

(71) Applicant : **CANON KABUSHIKI KAISHA**
30-2, 3-chome, Shimomaruko,
Ohta-ku
Tokyo (JP)

(72) Inventor : **Palmer, Douglas L.**
1 Calle Cabrillo
Foothill Ranch, California 92610 (US)
Inventor : **Ray, Richard Douglas**
28143 Via Fierro
Laguna Niguel, California 92677 (US)

(74) Representative : **Beresford, Keith Denis Lewis et al**
BERESFORD & Co.
2-5 Warwick Court
High Holborn
London WC1R 5DJ (GB)

(54) User-definable interactive system.

(57) Method and apparatus for creating, editing, and processing an interactive system made up of event-actuated action items. In one aspect, an event interpreter for selecting action items for execution based on occurrences of events in an interactive system includes an input section by which the event interpreter receives a computer-usable signal indicating that an event has occurred, an event name generator for generating an event name based on the computer-usable signal, and an application selector for comparing the event name to the event identifier for action items stored in a file system and for selecting for execution an action item whose event identifier corresponds to the event name. In another aspect, the action items of the interactive system are displayed hierarchically using the file system presenter/editor. In this aspect of the invention, an interactive system application containing at least one event-actuatable action item is selected from the file system, a graphical representation of the at least one event-actuatable action item is displayed in a hierarchical arrangement, and an event identifier associated with the displayed action item of the selected interactive system application is displayed. The associated event identifier actuates the action item in response to the occurrence of a physical event corresponding to the event identifier.

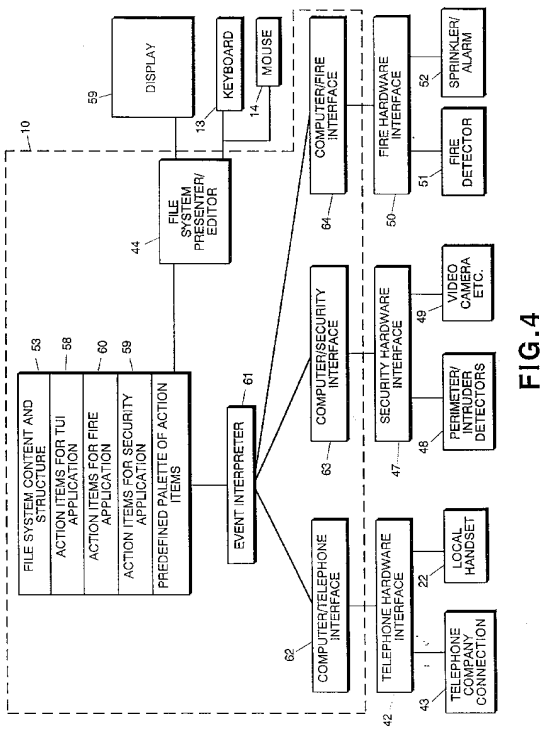

FIG.4

EP 0 657 810 A1

The present invention concerns a computer-implemented interactive system in which a computer can react to the occurrence of physical events by selecting and processing a stored action item so as to cause additional data processing or a controlled hardware response. More specifically, the invention is directed to an event interpreter which, based on the physical event, selects and processes an action item from a collection of action items which together form the interactive system, as well as to a method which uses the computer file system to create, modify and view the action items in the interactive system.

Computer-controlled interactive systems allow a computer to sense the occurrence of a physical event and to react to the physical event in accordance with a stored application of predefined responses, such as by performing specific data processing or by controlling hardware equipment. For example, in the case of an interactive voice response (IVR) system, a computer can sense the occurrence of an incoming call, respond by controlling the telephone to go off-hook and by playing out a series of user options, and can then sense the occurrence of DTMF tone commands from the caller and respond with appropriate data processing and telephone hardware control sequences.

Because of the complexity of interactive systems, they are ordinarily designed by highly trained software engineers using sophisticated and complicated software programming techniques. And, because the interactive systems must be created to an end user's specifications, most interactive systems are customized to that single user's requirements and cannot easily be tailored to other user's requirements. Thus, once an interactive system has been designed and coded, if changes are required, then either a new system must be written or the old system must be revised by highly trained software engineers at great expense.

Recently, some interactive systems have become available which allow their owner to make minor modifications on his own. However, in these cases the modification capability is itself a new program which the owner must learn and, once learned, it can only be used to modify the interactive system and not for any other purpose. Thus, current modification capabilities are specifically tailored for their associated interactive systems and cannot be used flexibly for other interactive systems.

The present invention addresses the above-noted drawback by allowing the owner to use the presenter/editor in the computer's file system, which is the same file system by which all files on the computer are created and modified and viewed, to create, modify and view the interactive system. According to this aspect of the invention, a method for visually representing content and structure of an interactive system application stored in a file system includes the steps of selecting from the file system an interactive system application containing at least one event-actuatable action item, displaying, in a hierarchical arrangement, a graphical representation of the at least one event-actuatable action item, and displaying an event identifier associated with the displayed action item of the selected interactive system application, wherein physical occurrence of an event corresponding to the event identifier causes actuation of the action item.

According to another aspect, the invention provides an event interpreter which controls processing of the interactive system by matching names of events to an event identifier of action items which comprise the interactive system, and by processing action items which are selected when a match is found. According to this aspect, an event interpreter for selecting action items for execution based on occurrences of events in an interactive system includes an input section by which the event interpreter receives a computer-usable signal indicating that a physical event has occurred, an event name generator for generating an event name based on the computer-usable signal, and an action item selector for comparing the event name to the event identifier for action items stored in a file system and for selecting for execution an action item whose event identifier corresponds to the event name.

According to another aspect of the present invention, an interactive system for monitoring and for responding to physical events includes a file system which includes a plurality of action items, each action item having an identifier corresponding to a physical event monitored in the interactive system. A physical event interface detecting an occurrence of a physical event and outputs a signal in response to the occurrence of an event. A name generator receives the signal from the physical event interface and generates an event name based on the received signal. An event interpreter selects and processes an action item from the file system, the action item having an event identifier corresponding to the generated event name.

In yet a further aspect of the invention, there is a method for visually representing content and structure of an interactive system using a conventional file system. In the method, an interactive system application containing at least one event-actuatable action item is selected from the file system. A graphical representation of the at least one event-actuatable action item is displayed in a hierarchical arrangement, and an event identifier associated with the at least one action item is displayed. The event identifier operates to actuate the at least one action item upon the occurrence of an event having a corresponding event name to the event identifier.

This brief summary of the invention is provided so that the nature of the invention may be understood quickly. A full understanding may be obtained by reference to the following detailed description of the invention in

connection with the appended drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a perspective view showing the outward appearance of an apparatus according to the present invention;
Figure 2 is a block diagram of the Figure 1 apparatus;
Figure 3 illustrates the hierarchy of the content and structure of the file system of the present invention;
Figure 4 is a representational view of an interactive system according to the present invention;
Figure 5a is a functional block diagram detailing the operation of the preferred embodiment of the event interpreters shown in Figure 3;
Figure 5b is a functional block diagram detailing the operation of an alternate embodiment of the event interpreter illustrated in Figure 3;
Figure 6, comprising Figures 6a and 6b, is a flow chart describing the method for responding to a physical event using the interactive system of the present invention;
Figure 6c is an example of the steps described in Figures 6a and 6b in the case of telephone user interface;
Figure 7 is an example of an action item tool palette for building a telephone user interface;
Figure 7a is an interactive service built by the tool palette in Figure 7;
Figure 8 is a flow chart describing the method for creating an interactive system using the tool palette shown in Figure 7;
Figure 9 is an example of a window-type display showing a telephone user interactive system and a security interactive system;
Figure 10 is a window-type view of a folder in the telephone user interactive system;
Figure 11, comprising Figures 11a and 11b, is a visual representation of a telephone user interactive system; and
Figure 12, comprising Figures 12a and 12b, is a visual representation of a security interactive system.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

Figure 1 is a view showing the outward appearance of a representative embodiment of the invention. Shown in Figure 1 is computing equipment 10, such as a MacIntosh, IBM PC, or PC-compatible computer. Computing equipment 10 includes a mass storage device such as disk drive 11 as well as display screen 12, such as a color monitor, keyboard 13 for entering text data and user commands, and pointing device 14, such as a mouse, for pointing to and manipulating objects displayed on screen 12.

Scanner 16 can be used to send image data to/from computing equipment 10. Data may also be input into computing equipment 10 from a variety of other sources such as a network via network connection line 23 or other sources via an unshown modem or facsimile. Printer 18 is provided for outputting documents produced by computing equipment 10.

Computing equipment 10 is connected to telephone handset 22 via an unshown telephone hardware interface (described below) and provides voice local operation over telephone lines 24. Telephone 22 is, in addition, another source of inputting/outputting voice data to/from computing equipment 10.

Computing equipment 10 is also connected to security hardware via security cabling 25, and to fire detection/prevention hardware via cabling 27.

It should be understood that although a programmable general purpose computer arrangement is shown in Figure 1, a dedicated or stand-alone computer or other type of data processing equipment can be used to practice the invention.

Figure 2 is a detailed block diagram showing the internal construction of computing equipment 10. As shown in Figure 2, computing equipment 10 includes a central processing unit (CPU) 30 interfaced with computer bus 31. Also interfaced with computer bus 31 is printer interface 32, scanner interface 33, network interface 34, display interface 35, keyboard 36, pointing device 37, main memory 40, disk 11, telephone hardware interface 42, security hardware interface 47, and fire hardware interface 50.

Main memory 40 interfaces with computer bus 31 so as to provide a random access memory storage and read-only memory storage for use by CPU 31 when executing stored program instructions. More specifically, CPU 30 loads process steps from disk 11 into main memory 40 and executes the process steps out of main memory 40.

Disk 11 has stored thereon both stored program instruction sequences which are processed by CPU 30 as well as data files stored hierarchically. More specifically, and as shown in Figure 2, disk 11 stores a "file system" 53 such as a MacIntosh file system, a DOS file system, a customized file system, or any comparable

system, such as a database. "File system" means the portion of the computer operating system by which files are created and physically located and stored on storage media and includes, for example, conventional file systems such as a MacIntosh hierarchical file system or a Microsoft Windows DOS file system, and it also includes customized file system 53 which creates and physically locates and stores files on media, such as disk 11.

As shown in Figure 2, file system 53 includes file system presenter/editor 44 which provides a graphical user interface to the file system stored on disk 11 and allows a user to create, edit, and manipulate the content and structure of the file system. In addition to creating, editing, and manipulating the content and structure of the file system, file system presenter/editor 44 allows an operator to run applications directly and to generate a visual representation of the file system content and structure via a graphical user interface displayed on screen 12.

The content and structure of the file system includes a variety of software structures consisting primarily of directories, stored application programs 54 which include program instruction sequences executed by CPU 30 to control operation of computing equipment 10, as well as stored data 55 which is manipulated, processed, and displayed by CPU 30 in accordance with the stored program instruction sequences.

One example of application programs stored on disk 11 is message manager application program 56 by which an operator can create, manipulate, view, send, and receive multimedia messages via modem or facsimile over telephone line 24 or over network connection 23. Multimedia messages 57 may contain a variety of objects such as text objects, bit map image objects, computer graphics objects, sound (such as voice or MIDI music) objects, and motion video objects. The message manager application utilizes functions of a telephone user interface application program (discussed below) for sending and receiving voice messages as well as facsimile and modem messages. For example, an operator creates a voice message using the message manager application and sends the voice message to an intended receiver via an outside telephone line. In order to send the message, the telephone user interface application program operates to send the voice message created by the message manager application to the designated telephone number using various process steps. Similarly, upon receiving messages, the telephone user interface application program operates to record voice messages and to place each message in a voice folder using various process steps. Once a message has been received, the message manager application operates to identify the recipient of each received voice message and stores a listing of the received messages in an "In-box" for an operator's retrieval.

Other application programs and their associated data may also be stored on disk 11, such as word processing application programs, spreadsheet programs, communication programs, and similar data processing programs.

The content and structure of the file system also includes application programs for the interactive systems which are the subject of the present invention, here shown representationally as telephone user interface ("TUI") application program 58, security application program 59, and fire application program 60. These application programs are each organized in "action items" which are described more fully below (see "Organization Of Action Items"). Generally, each interactive system responds to an event, such as detection of a telephone ringing signal or a DTMF tone, or detection of a perimeter breach, or detection of excessive smoke, or a software-created event, by selecting and processing an action item in the interactive system. The specific action item which is selected is determined by event interpreter 61 based on a match between an event name of the physical event and event identifiers for the action items. The event interpreter then causes the selected action item to be processed, which in turn may cause other applications to be activated and processed, and hardware to be controlled.

As shown in Figure 4, software interface between computing equipment 10 and the telephone, fire, and security hardware (42, 47, and 50, respectively) is provided by computer/telephone interface 62, computer/security interface 63, and computer/fire interface 64, respectively. These software/hardware interface process interrupts from hardware devices 42, 47, and 50, the interrupts indicating that physical events have occurred, and translate the interrupts into a computer format usable by event interpreter 61 and/or other application programs on disk 11. Conversely, these software/hardware interfaces convert computer commands generated by the action items (and/or other application programs) into a format usable by the hardware devices 42, 47, and 50, whereby computing equipment 10 is able to effect control over the hardware devices.

In operation, a user may, for example, activate TUI application program 58. When telephone hardware interface 42 receives an incoming call via telephone company connector 43 or local telephone handset 22, an interrupt from telephone hardware interface 42 is received by computer bus 31, at which point computer/telephone interface 62 is retrieved from disk 11 and stored in main memory 40 by CPU 30 to process the hardware signal. The hardware signal, in accordance with the process steps of computer/telephone interface 62, generates an event identifier corresponding to the signal.

Event interpreter 61 utilizes the generated event name to locate an action item in file system 53 which has

a corresponding event identifier. Upon locating an action item with the corresponding event identifier, event interpreter 61 processes the action item, which either generates a response to the physical event or which operates to process further action items which ultimately generate a response to the physical event. In either case, the generated response may be communicated back through computer bus 31 to computer/telephone interface 62 or communicated to other system components such as various output interfaces in the system. Computer/telephone interface 62 outputs the response to telephone hardware interface 42. Telephone hardware interface 42 generates various audible responses to the source of the incoming call, which could be telephone company connector 43 or local telephone handset 22.

(Hierarchical Organization Of Action Items)

Figure 3 illustrates the hierarchy of the content and structure of file system 53 which in the present case is arranged by folders, documents, applications, and links (also known as directories, data files, programs, and aliases, respectively). In the present invention, each folder, document, application, and link represents an "action item" which collectively define an interactive system.

Thus, as shown in Figure 3, a telephone interactive service comprises a hierarchical structure which contains folders, documents, programs, and links. These items have designatable names which represent a physical event which actuates the item. As shown in Figure 3, the interactive system is hierarchically arranged in a sequence of folders. Each folder, starting from a root folder, can contain other folders and, in addition, can also contain documents, applications, and links. Thus, root folder 71 contains incoming call folder 72. Incoming call folder 72 contains therein voice folder 73 and facsimile folder 74. Within voice folder 73 and facsimile folder 74 there are documents, folders, and applications.

As stated above, each folder, document, application, and link is an "action item". Each action item has a predefined function which in some cases performs a predefined task. In addition, each action item has an "event identifier" by which the action item may be actuated. For example, a folder action item is opened when its event identifier corresponds to a detected physical event and, likewise, any items in the opened folder are processed according to an activating event. In some instances, the entire folder will be processed, since each action item therein has been actuated by the initial actuating physical event. In the case the action item is an application, process steps in the application will be executed. On the other hand, if the action item is a document containing data such as image data, the type of data is identified and respective data processing steps are used to process the data therein. In the case of a link, the link action item will direct processing to another action item, such as a folder, a data item, etc.

As described above, an event identifier of an action item is used to actuate the action item. Thus, as shown in Figure 3, in the case of an incoming telephone call, the event name "incoming call" matches the event identifier of the root folder and, therefore, root folder 71 is retrieved from file system 53 to be processed. Upon retrieving the "incoming call" root folder 71, root folder 71 is opened.

(Construction Of Interactive Systems)

Figure 4 is a functional block diagram for explaining the relation and function of the various software and hardware components of an interactive system according to the present invention.

The present invention, as described above, is implemented with a telephone user interactive system, a security interactive system, and a fire detecting interactive system. It is to be understood that such are merely examples of the types of interactive systems which can be used with the present invention and that the present invention is not limited to the examples described herein. In addition, the present invention responds to physical events which may be externally occurring events, such as an incoming telephone call, and internally occurring events, such as elapsed time, and computer-generated events, such as opening a folder.

As shown in Figure 4, telephone hardware interface 42, such as the type described in U. S. Patent Application Serial No._____, filed _____, and entitled "Two-line Telephone Controller", transmits/receives telephone signals to/from telephone company connector 43 and local handset 22. Security hardware interface 47 receives signals from perimeter/intruder detectors 48 and, in response, security hardware interface is instructed to send signals to security devices 49 such as video cameras, audible and silent alarms, and alerting law enforcement authorities. Fire hardware interface 50 receives signals from fire and smoke detectors 51 and, in response, fire hardware interface 50 is instructed to activate the sprinkler/alarm system 52.

For each hardware interface 42, 47, and 50, there is provided a corresponding computer software interface such as the aforementioned computer/telephone interface 62, computer/security interface 63, and computer-/fire interface 64. Each computer software interface 62, 63, and 64 receives and converts signals to/from its

respective hardware interface. In this regard, the manner by which computer software interfaces 62, 63, and 64 receive signals from respective hardware interfaces may be by an interrupt driver or may be by polling/querying each respective hardware interface after a predetermined elapsed time period.

As shown in more detail in Figure 5a, each computer software interface 62, 63, and 64 includes event detector 75, event name mapping 76, and hardware controller 82.

Event detector 75 receives event signals from its respective hardware interface and compares the received signal to event names in event name mapping 76. Mapping of occurrences of physical events to an event name is provided by event name mapping 76. Preferably, event name mapping 76 includes event names which bear an english language indication of the physical event, such as "ringing" when an incoming telephone call is detected by telephone hardware interface 42, "DTMF=9" when telephone hardware interface 42 detects modulated tones indicating that a touch-tone key "9" has been depressed, "zone breach" when security hardware interface 47 detects a perimeter breach, or "fire" when fire hardware interface detects smoke or heat. The generation of event names need not be generated by the computer software interface but could be generated by event interpreter 61 in much the same way.

The following table provides examples of events which are monitored in a telephone interactive service.

Telephone Events

Event	Description	Event Name
INCOMING	Incoming Call	(INCOMING)
DTMF	Remote DTMF tones	(DTMF=n) Where n=DTMF tone(s) to trigger event. n may be a string of DTMF tones. If (DTMF = NONE) no matching event was found for received DTMF tone.
LOOP	Iterations waiting for DTMF tones. When a timeout occurs waiting for DTMF (ie, no DTMF tones were received from caller) a counter is incremented. When this counter is equal to the LOOP argument the action is processed. The timeout for DTMF is equal to five seconds.	(LOOP=n) Where n is the number of DTMF timeouts to occur before action is processed.

ENTER	Folder Access. When a folder is "entered" from above.	(ENTER)
ALWAYS	Any event	(ALWAYS)
CALLERID	Caller's phone number is matched	(CALLERID=n) Where n is the callers phone number will be unformatted numbers (ie (714)221-1234 will be 7142211234)
HANDSET OFF HOOK	Local handset taken off hook	This is a built-in event that is globally available and is not represented on the desktop
FAX TONE	Fax tone received	(FACSIMILE)
SILENCE	DTMF timeout occurred at first level folder after any LOOP events were processed	(SILENCE)
ERROR	An unrecoverable error occurred in IVR application	This is a built-in event that is globally available and is not represented on the desktop

Upon mapping the physical event to its event name, event detector 70 outputs the name to event interpreter 61. Event interpreter 61 matches event names to event identifiers for the action items in the interactive systems. When a match is found, the event interpreter selects the matching action item and processes it. In connection with processing the action associated to the event, event interpreter 61 may transmit a response signal back through a computer software interface to hardware controller 82. Hardware controller 82 sends the signal to the designated hardware device, which performs the action. Processing of the action item may also entail data processing or computer execution of an application, such as the message manager application. These are all described in more detail below.

(Event Interpreter)

Figure 5a shows details of event interpreter 61. As shown in Figure 5a, event interpreter 61 includes name parser 77, event name/action item list 78, event matcher 79, action item selector 80, and action item execution processor 81.

Upon initialization, name parser 77 opens the interactive system's root folder, gathers the event identifiers of each action item therein, and places the event identifiers into event-name/action item list 78. Because event-name/action item list 78 maintains a current listing of parsed names from the selected folder, event interpreter 61 does not return to file system 53 each time in order to make a comparison. In this manner, event interpreter 61 operates more efficiently. Of course, if event-name/action item list 78 is not provided, the system will still

operate but with less efficiency.

Once the system is in operation, name parser 77 operates much in the same way as during initialization, however, name parser 77 opens the currently selected folder and gathers and stores event identifiers of each action item. As soon as the event-name/action list 78 has been created, event interpreter 61 monitors and responds to signals from each computer software interface.

Event matcher 79 receives input event names generated by each computer software interface 62, 63, and 64. Event matcher 79 compares the event name to the event-name/action item list 78. In some instances, the event identifier having the corresponding event name can not be located, and in such a situation the system merely ignores the input event name and continues to monitor for events.

On the other hand, when an action item event identifier matches the event name, action item selector 80 uses the event identifier to locate and retrieve the action item stored in file system 53 for processing. After action item selector 80 retrieves the action item, action item execution-processor 81 processes the action item. The processed action item may result in further data processing which requires action item execution-processor 81 to output an event name to event matcher 79 so that another action can be retrieved from file system 53 or, when the result of processing the action item generates a controlled hardware response, action item execution-processor 81 transmits a response signal either to the computer software interface from which the event name was originally generated or to an appropriate system component, depending on the process steps in the processed action item. The controlled hardware response signal is processed by hardware controller 82 in the respective computer software interface.

Hardware controller 82 directs a hardware command to the designated hardware recipient via the hardware interface. For example, in the case of a zone being tripped in the security interactive service, event interpreter 61 processes the associated action items which generate a controlled hardware response such as activate audible alarm and activate video cameras. This hardware response signal is received by hardware controller 82 which in turn, transmits actuation signals to the alarm and video cameras via security hardware interface 47.

As shown in 5b, an alternate embodiment of event interpreter 61 includes name parsers 77a, 77b, and 77c, event-name/action item list 78a, 78b, and 78c, event matcher 79, action item selector 80, and action item execution processor 81.

In the alternate embodiment, there is provided a name parser and event-name/action item list corresponding to each interactive system in the interactive system. As a result, event identifiers in the root folder for each interactive system are parsed by its corresponding name parser which, in the present case, is telephone service name parser 77a, security name parser 77b, and fire name parser 77c. Each name parser 77a, 77b, and 77c provides parsed event identifiers for the creation of a corresponding event-name/action item list which, in the present case, is telephone event-name/action item list 78a, security event-name/action item list 78b, and fire event-name/action item list 78c.

Once each event-name/action item list is created, event matcher 79 matches received event names from computer software interfaces to each event-name/action item list 78a, 78b, and 78c, and, upon locating a matching event identifier, action item selector 80 selects an associated action item from file system 53. After retrieving the selected action item, the alternate embodiment of event interpreter 61 functions much in the same manner as the preferred embodiment of event interpreter 61.

The alternate embodiment of event interpreter 61 operates more efficiently to parse and process a current folder for each interactive system. That is, the alternate embodiment of event interpreter 61 does not have to interrupt a folder currently being processed so as to react to a physical event occurring in another interactive service. Since current folders for each interactive service have been parsed by its respective name parser, event matcher 76 can match event names to each event-name/action list for each interactive service without having to interrupt and retrieve another folder to be parsed, and without the possibility of ignoring an event name from one interactive system because only event identifiers from another system are stored in list 78.

The above-described alternate embodiment is only one example for increasing the efficiency of the interactive system. Other alternate embodiments, such as using multiple event interpreters in the interactive system, could be used as an alternative without deviating from the scope of the present invention.

Figures 6a and 6b comprise a flow chart illustrating the operation of the interactive telephone user interface system described in Figure 3.

Upon initialization, the systems root folder is automatically retrieved and parsed by name parser 77 in step S601. In step S602, the parsed names are used to create event-name/action items list 78. Once event-name/action item list 78 is created, event interpreter 61 monitors each system interface in step S603. Event interpreter 61 monitors each interface either by querying each interactive interface after a predetermined elapsed time or by being interrupted by an interrupt signal sent by the system interface in step S604. In the case that no event has occurred after each interface has been queried or if no interrupt has been received by

event interpreter 61, flow returns to in step S603.

On the other hand, in a case that an event has occurred, for example, computer telephone interface 54 receives an incoming call, flow advances to step S605. In step S605, the appropriate one of computer interfaces 62, 63, or 64 identifies the physical event and designates an event name based on a comparison of the received signal to an event name map. In step S606, event matcher 79 compares the event name to event identifiers in event-name action item list 78.

In step S607, event matcher 79 determines if a match has been located between the event name and event identifiers in list 78. If no match can be made in step S607, the physical event is ignored and flow returns to step S603 where event interpreter 61 continues to monitor the interactive system. On the other hand, if event matcher 79 matches the event name to an event identifier of a folder, document, application, or link, the corresponding action item is retrieved from file system 53 by action item selector 80 (step S608).

In step S610, action item execution-processor 81 processes the contents of the action item having the corresponding physical event name. If the action item is a folder (step S611), event interpreter 61 opens the folder and flow returns to step S601 where the event identifiers for the action items in the newly opened (and now current) folder are parsed and replace the identifiers stored in list 78. In step S612, a controlled hardware response signal may be generated by action item execution-processor 81 based on the processed steps in the action item, and a signal is output to the computer software interface which in turn sends the signal to the hardware interface in order to produce a hardware response to the physical event.

Figure 6c is an example of the above steps in the case of a telephone user interface (TUI) interactive system. In the example illustrated in Figure 6c, the computer telephone interface 62 identifies the event as an "incoming call" and sends an event name to event interpreter 61. Event matcher 79 compares the event name to event identifiers in list 78. Upon matching the event name, action item selector 80 locates an action item having an event identifier corresponding to "incoming call" in file system 53. Upon locating the "incoming call" action item, the action item is retrieved from the file system.

Since the "incoming call" action item is a folder, action item execution-processor 81 "opens" the "incoming call" folder. The event identifiers in that folder are then parsed and stored to replace the identifiers in list 78. Here, all identifiers are "always", which is a computer-generated event name. Thus, the "greeting" action item is named "always". Pre-recorded voice data document is retrieved from file system 53, processed by action item execution-processor 81, and the result is output to the caller via computer/telephone interface 62. After playing the greeting, event interpreter 61 will execute the next action item with the same event identifier in left-to-right order. If appropriate, the action item will be processed, or the event interpreter 61 will continue to monitor for incoming events.

In the present case, the next action item is an "always" item and, therefore, it begins to record a caller's message input through telephone hardware interface 42.

(Visual Representation Of The Content And Structure Of The File System)

Reverting to Figure 4, file system presenter/editor 44 is a component of the computer's operating system which provides a graphical user interface of file system 53. File system presenter/editor 44 allows a user to open, view, edit, create, and delete the structure and content of file system 53, including any of the action items in the interactive systems. File system presenter/editor 44 represents through a graphical user interface the content and structure of file system 53 and allows the user to manipulate the names and attributes of a folder, document, application, and link which correspond to action items in file system 53.

The graphical user interface is a representation of the content and structure of file system 53. Various representations can be used for graphical user interface such as MacIntosh's Finder which uses a folder/document/application hierarchical representation, or Windows' FileManager which uses a directory/file/program hierarchical arrangement or any custom graphical user interface which gives a depiction of the content and structure of the file system.

In the present invention, the graphical user interface defined by file system presenter/editor 44, graphically represents the content of a folder as a window with icons within the window. These icons represent action items which correspond to folders, documents, applications, and links in file system 53. File system presenter/editor 44 also presents visible text (the aforementioned event identifier) with each displayed icon in the graphical user interface.

The graphical user interface provides the user with a hierarchical view of file system 53 so that the user intuitively understands its structure. In addition, the graphical user interface displays user-designatable menu options which will affect currently highlighted icons. For example, one menu option will permit viewing and editing of attributes of the currently selected action item or renaming of an action item. "Clicking" on a folder icon with mouse 14 will open a window that shows the contents of the folder. "Double clicking" on a document icon

will execute the action item's associated process steps (editor) to process data within the document. For example, double clicking on a voice document will execute process steps for storing, editing, and manipulating voice data and double clicking on an image document will execute process steps for storing, editing, and manipulating image data in that document.

File system presenter/editor 44 interprets user selections, via keyboard 13 or mouse 14, and permits opening, viewing, and editing of files. File system presenter/editor 44 also obtains information about the content and structure from file system 53 and utilizes a windowing environment interface to represent, hierarchically, the content and structure of file system 53 visually to a user. In this manner, a user can intuitively manipulate and create action items within file system 53. While windowing environments are believed well-suited for hierarchically representing the content and structure, other environments are also usable, for example, a tree-like representation.

A discussion of the method of creating and visually displaying an interactive system will be discussed in detail below with respect to Figure 7.

Figure 7 shows tool palette 90 which aids the user in building an interactive system. Tool palette 90 includes functions to create, name, and edit action items in a currently selected folder. In the example shown in Figure 7, tool palette 90 contains action items to build a telephone user interface (TUI). Within tool palette 90, there is tool bar 91 of predefined action items which are useful in building the interactive system. Tool bar 91 contains icons representative of the various types of action items. Each action item performs a predetermined task useful in building a TUI such as playing a recorded voice message through a telephone line, recording a voice message, or sending a document by facsimile.

Event window 93 within tool palette 90 contains a listing of predefined event identifiers useful in building the interactive system. Tool palette 90 includes functions to create, name, and edit action items in a currently selected folder. Preferably, the event window contains the names of all the events in mapping 76. Once a user has selected an action item from tool bar 91, the event window 93 permits the user to name the selected action item with a respective physical event identifier.

Value window 94 is used to input additional parameters which are used to form or to complete the event specification which actuates the action item. Comment window 95 allows the user to add additional descriptive text which may include a further description of the action item. In this regard, the added text is ignored by event interpreter 61 when selecting and executing the action item or folder.

As described above, folders, documents, applications, and links are standard items provided by file system 53 and graphically represented by file system presenter/editor 44. The action items within a folder provide an executable action in response to a physical event. Each folder, document, application, and link has an event identifier and an attribute set by the user. The event identifier and attributes can be changed by the user via the graphical user interface (tool palette 90) and the presenter/editor. Attribute values determine whether the action item is an input response or an output response. The attributes can be stored with each action item or stored separately from the action item. For example, in the case of the telephone user interface system, if a data document of ASCII text has a "speak" attribute, it may be played, via telephone hardware interface 42, using text-to-speech methods.

To create an interactive system, the user selects from tool bar 91 any of the action items by clicking on the graphical representation or icon with mouse 14. In a TUI, for example, the system designer would create an interactive system by selecting various icons from tool bar 91 so as to answer an incoming call, play a greeting, play a recorded menu of possible selections, request input of a selection using DTMF tones, output selected information, terminate the call, etc.

The table below includes descriptions of some of the action items contained in tool bar 91 illustrated in Figure 7.

EXAMPLES OF ACTION ITEMS IN FIG. 7

NAME OF ACTION	DESCRIPTION	ATTRIBUTE
Folder (98)	Make the folder the current folder.	None
Play Voice (96)	Play a voice document.	Send
Record Voice (99)	Record voice off phone line to a voice object.	Receive
Send Message (97)	Send a message.	Send
End The Call (102)	End the call by hanging up.	Send
Say Number Of Messages (100a)	Use concatenated speech to tell the caller the number of messages in a folder.	Send
Say Message Info (100b)	Use concatenated speech to tell the caller the date, time, and method of delivery of the currently selected message in a folder.	Send
Say Last DTMF (100c)	Use concatenated speech to speak the stored DTMF digits to the caller.	Send
Say Current Date And Time (100d)	Tell caller the current date and time using concatenated speech.	Send
Say Elapsed Time (100e)	Tell the caller the elapsed time he has been on the line using concatenated speech.	Send

Skip To Next Message (101b)	Make the current message the next one in a specified folder.	None
Play Message (101a)	Play a voice message to the caller.	Send
Forward Message (101c)	Forward message to a specified folder.	
Delete Message (101d)	Delete current message.	

Figure 7a illustrates a basic interactive telephone user interface (TUI) system for answering an incoming telephone call, recording a caller's message, playing a recorded farewell, and terminating the phone call, and Figure 8 is a flow diagram for explaining how a user uses tool palette 90 to create this TUI.

Upon entering a request to create a telephone user interface, file system presenter/editor 44 retrieves a telephone service builder tool palette, such as tool palette 90. In step S801, file system presenter/editor 44 displays tool palette 90, via the graphical user interface, which includes graphical representations of user-selectable action items. In step S803, the user creates a folder to be used to store various action items selected by the user.

To create an interactive service as shown in Figure 7a, a user selects a name for the action item to be selected using event window 93 (step S804) and selects an action item from tool bar 91 by clicking on the desired action item. Upon clicking on the desired action item, the selected action item appears in the folder (step S805) with its event identifier which actuates the action item (as shown in brackets at reference number 107). As shown in Figure 7a, action item 96 has been selected and has been created in folder 105 as shown at 106 with its event identifier. In step S806, the user customizes the action item by entering commentary via comment window 95 (as shown at reference number 108). In the present example, action item 106 has been assigned an event identifier of "always", meaning that the action will be executed every time folder 105 has been entered and has been defined by the comments as a "greeting". In addition, the event identifier shown at reference number 107 can be changed by entering a new name in event window 93 while the action item is currently selected.

The user builds the entire interactive system program by naming, selecting, and entering commentary (if necessary), for each action item in the interactive service. An action item can be customized further by double clicking on the action item icon to execute the associated editor to further define the behavior of the action item application or to create, edit, or manipulate data in an action item document.

For example, to record the greeting message associated with action item icon 106, the user double clicks the icon so as to open the action item. The file system presenter/editor 44 then permits the user to record a voice greeting message, e.g., using handset 22. Presenter/editor 44 automatically allows voice recording when icon 106 is double-clicked because of the attributes that are stored with predefined action item 96 in tool bar 91. The other predefined action items in tool bar 91 have appropriate attributes stored with them which allow, for example, entry of text for a text item, designation of file space for a voice recording action item, etc.

The user continues to build the interactive TUI by selecting action item icons. Thus, as shown in Figure 7a, the user has selected a message recorder icon 99 at 106a, and has assigned an appropriate event identifier, all as described above (see steps S804, S806, and S807).

Step S803 through step S807 are repeated until the interactive service is complete. File system presenter/editor 44 displays the service as it is being created or manipulated by using retrieved visual representations of action items in step S809.

Once a user has initially created a portion of an interactive system, the user may use existing folders, documents, and application items by copying and altering the content and structure of the item. It is also possible to create various interactive services by (1) further customizing an existing interactive service, (2) building an interactive service from predefined, linked action items (i.e., folders having an arrangement of action items therein to perform predetermined tasks), (3) building an interactive service from newly created action items,

or (4) building an interactive service from predefined, individual action items as described above. Because the interactive system can be intuitively designed using graphical representations of the hierarchy and content of the system, a user can design an interactive system program with precision and with ease.

As shown in Figure 7a, file system presenter/editor 64 presents the content of folder "simple answer" as "action item" icons having visible text names which relate to physical events which actuate the item in the interactive system.

The method of visually representing interactive service application using the present invention will be described in greater detail by way of the examples illustrated in Figures 9-12.

The interactive system illustrated in Figure 4 includes a telephone interactive service, a security interactive service, and a fire detector interactive service. As shown in Figure 9, file system 53 includes a telephone interactive service root folder 109 and a security interactive service root folder 110 (the fire detector interactive service folder is included in the security interactive service root folder 110) which include action items to respond to events in each respective interactive service.

As described above, during operation of, say, the telephone user interactive system, when an "incoming call" event occurs, computer telephone interface 62 sends an event name to event interpreter 61. Event interpreter 61 locates action item folder "telephone service" 109 in file system 53. Upon locating action item folder "telephone" 109 shown in Figure 9, action item folder 113 is entered for further processing.

Visual representation of this interaction is shown in Figure 10. As seen there, "telephone service" folder 109 is displayed by file system presenter/editor 44 as a window folder having three action items stored therein which are indicated by reference numerals 114, 115, and 116. The compound event identifier of action item 114 indicates that action item folder 114 is opened only if both conditions are true. In this case, the physical events which open action item 114 are a voice transmission and the time of day being between 8 p.m. and 9 a.m. Action item folder 115 is a facsimile folder which is processed if the incoming call is a facsimile transmission and action item 116 responds to a compound event which corresponds to a voice transmission and a time of day being between 9 a.m. and 8 p.m.

The contents of folder 114 are visually displayed by file system presenter/editor 44 as illustrated in Figure 11a. Folder 114 is depicted as containing various types of action items. For example, action item 126 represents an application which plays a pre-recorded greeting upon entering folder 114, and action item 127 represents an application which plays a pre-recorded menu of selections by which a caller can select various functions of the service by depressing touch-tone keys on the caller's telephone. For example, "DTMF = 1" represents an action item folder 130 which is activated upon depression of a touch-tone button "1". In addition to the above-described action items, call termination action item 135 is provided to terminate a telephone call upon an elapsed time of three seconds or upon a caller's hang-up.

As shown in Figure 12a, file system presenter/editor 44 visually represents security interactive service root folder 110 as having various action items and folders. In folder 110, action items 141 and 142 represent application action items which are processed in response to an internally monitored event such as the present time of day being between 7 a.m. and 7 p.m. For example, action item 141 may be used to disarm the security system and action item 142 may be used to arm the security system if the time of day is between 7 p.m. and 7 a.m. On the other hand, folders 144 and 145 represent folder action items which respond to externally monitored physical events monitored by security interface 47.

As shown in Figure 12b, action item folder 143 is visually represented as having action items which respond to events occurring in a secured perimeter area. As shown in Figure 12b, file system presenter/editor 44 visually represents "perimeter tripped" folder 143. For example, "always" action item 147 represents an action which automatically locks the safe, "always" action item 148 represents an action which actuates the audible alarm, and "always" action item 149 represents an action which automatically alerts the police.

The above descriptions of the telephone user interactive system and the security interactive systems are merely examples of some of the visual representations of interactive services which can be generated and viewed hierarchically by file system presenter/editor 44. It is to be understood that any internally monitored event and any externally monitored event can be substituted for the ones described herein. For example, an interactive system could be designed to monitor physical events in a robotic assembly line manufacturing plant, a medical system, or any other type of event-responsive interactive system.

The US patent application entitled "2-line telephone controller" referred to above has a corresponding European patent application (agent's ref 2336230) of the same title, filed 2 December 1994. The content of each of these applications is incorporated herein by reference.

Claims

1. An event interpreter for selecting action items for execution based on occurrences of events in an interactive system, comprising:
an input section by which the event interpreter receives a computer-usable signal indicating that an event has occurred;
an event name generator for generating an event name based on the computer-usable signal; and
an action item selector for comparing the event name to the event identifier for action items stored in a file system and for selecting for execution an action item whose event identifier corresponds to the event name.

2. An event interpreter according to Claim 1, wherein the input section includes at least one hardware event interface for monitoring signals from a plurality of hardware event detectors.

3. An event interpreter according to Claim 2, wherein the at least one hardware event interface monitors a telephone system.

4. An event interpreter according to Claim 2, wherein the at least one hardware event interface monitors a security and fire detection system.

5. An event interpreter according to Claim 1, wherein the event name generator converts the signal from the input section by mapping the signal to event names in an event-name map.

6. An event interpreter according to Claim 1, wherein the computer-usable signal is generated by a computer software interface which receives an input signal from the hardware event interface.

7. An event interpreter according to Claim 1, further comprising an action item processor for processing the selected action item and, in accordance with the result of processing, for producing a controlled hardware response signal to the occurrence of the physical event.

8. An event interpreter according to Claim 1, further comprising a display screen for displaying a graphical user interface which graphically displays the action item, and wherein the graphical user interface is used to manipulate and edit the displayed action item.

9. An interactive system for monitoring and for responding to physical events, comprising:
a file system which includes a plurality of action items, each action item having an identifier corresponding to at least one physical event monitored in the interactive system;
a physical event interface for detecting an occurrence of at least one physical event and for outputting a signal in response to the occurrence of an event;
a name generator for receiving the signal from the physical event interface and for generating an event name based on the received signal; and
an event interpreter for selecting and for processing an action item from the file system, the action item having an event identifier corresponding to the generated event name.

10. An interactive system according to Claim 9, further comprising a physical event detector for detecting a physical event and outputting an event detection signal to the physical event interface upon the occurrence of the physical event.

11. An interactive system according to Claim 10, wherein the name generator generates an event name by mapping the received signal into an event-name map of event names.

12. An interactive system according to Claim 10, wherein the physical event detector detects physical events in a telephone system, and wherein, as a result of the occurrence of the physical event, the event interpreter causes a physical response to be produced based on the processed action item.

13. An interactive system according to Claim 10, wherein the physical event detector detects physical events in a security system, and wherein, as a result of the occurrence of the physical event, the event interpreter causes a physical response to be produced based on the processed action item.

14. An interactive system according to Claim 9, further comprising an action item builder for creating and edit-

ing action items to be stored in the file system, wherein the action item builder provides event identifiers for each action item by which the action item is actuated.

15. An interactive system according to Claim 9, wherein the action items to be processed include at least one of voice data, text data, and image data.

16. An interactive system according to Claim 9, wherein action items are comprised by a folder, a document, or an application.

17. A method for displaying a graphical representation of an event-based system, said method comprising the steps of:

selecting a folder containing at least one event-actuatable action item;

displaying, within a window corresponding to the selected folder action item, an icon representative of the function of the at least one event-actuatable action item; and

displaying, in conjunction with the icon, an event identifier which causes actuation of the action item.

18. A method according to Claim 17, further comprising the step of designating the displayed icon and, in response to designation, permitting editing of the action item represented by the icon.

19. A method according to Claim 17, further comprising the steps of designating the displayed icon and, in response to designation, actuating the action item represented by the icon.

20. The method according to Claim 17, further comprising the steps of:

displaying, within the window corresponding to the selected folder, at least one folder action item; and

displaying, in conjunction with the folder action item, an event identifier which causes the folder to be accessed.

21. The method according to Claim 17, further comprising the steps of:

displaying, within the window corresponding to the selected folder, at least one folder action item and at least one executable action item representative of an event-actuatable application; and

displaying, in conjunction with the folder action item and with the executable action item, respective event identifiers which cause the folder to be accessed and the executable action item to be processed.

22. A method according to Claim 17, wherein the selected folder contains additional folder action items and action items which represent event-actuatable applications.

23. A method according to Claim 17, wherein the displayed action item is a graphical representation of a physical response to the detected event.

24. A method according to Claim 17, wherein the displayed identifier is representative of the name of a corresponding actuating event.

25. A method according to Claim 17, wherein the steps of displaying comprises displaying a graphical user interface using a file system/presenter editor whereby action items may be created, edited, opened, named, copied, or deleted displayed action items.

26. A method according to Claim 17, wherein the displayed window and its contents represent an interactive system.

27. A method for creating a user-definable interactive system using a tool palette of predefined action items, comprising the steps of:

storing a plurality of action items, each of the plurality of action items performing a predetermined task;

displaying a tool palette of graphical representations corresponding to each of the plurality of action items;

responding to a designation of one of the displayed graphical representations by selecting the corresponding action item; and

associating graphical representations of each of the selected action items so as to define at least

part of the interactive system.

28. A method according to Claim 27, further comprising the step of selecting an event identifier by which the action item is actuated.

29. A method according to Claim 27, wherein the tool palette and the associated graphical representations of each of the selected action items are displayed in a windowing environment.

30. A method according to Claim 27, wherein the responding step responds to a user designation of a graphical representation with a pointing device and displays the designated action item in a currently selected folder.

31. A method according to Claim 27, wherein the displaying step displays icons representative of a physical response which is executed upon actuation of the action items.

32. A method according to Claim 27, wherein the steps of displaying comprises displaying using a file system presenter/editor whereby action items may be created, edited, opened, named, copied, or deleted displayed action items.

33. A method for visually representing content and structure of an interactive system application stored in a file system, comprising the steps of:

selecting from the file system an interactive system application containing at least one event-actuatable action item;

displaying, in a hierarchical arrangement, a graphical representation of the at least one event-actuatable action item; and

displaying an event identifier associated with the displayed action item of the selected interactive system application, wherein physical occurrences of an event corresponding to the event identifier causes actuation of the action item.

34. A method of or apparatus for constructing a re-configurable interactive system for implementation by a computer, wherein a hierarchical file system of the computer is used to define a hierarchical structure for the interactive system.

35. A method or apparatus according to claim 34, wherein the interactive system comprises an event interpreter for traversing levels of the hierarchical structure in accordance with physical events.

36. An event interpreter suitable for use in a method or apparatus of any preceding claim.

37. A method or apparatus according to any combination of the preceding claims.

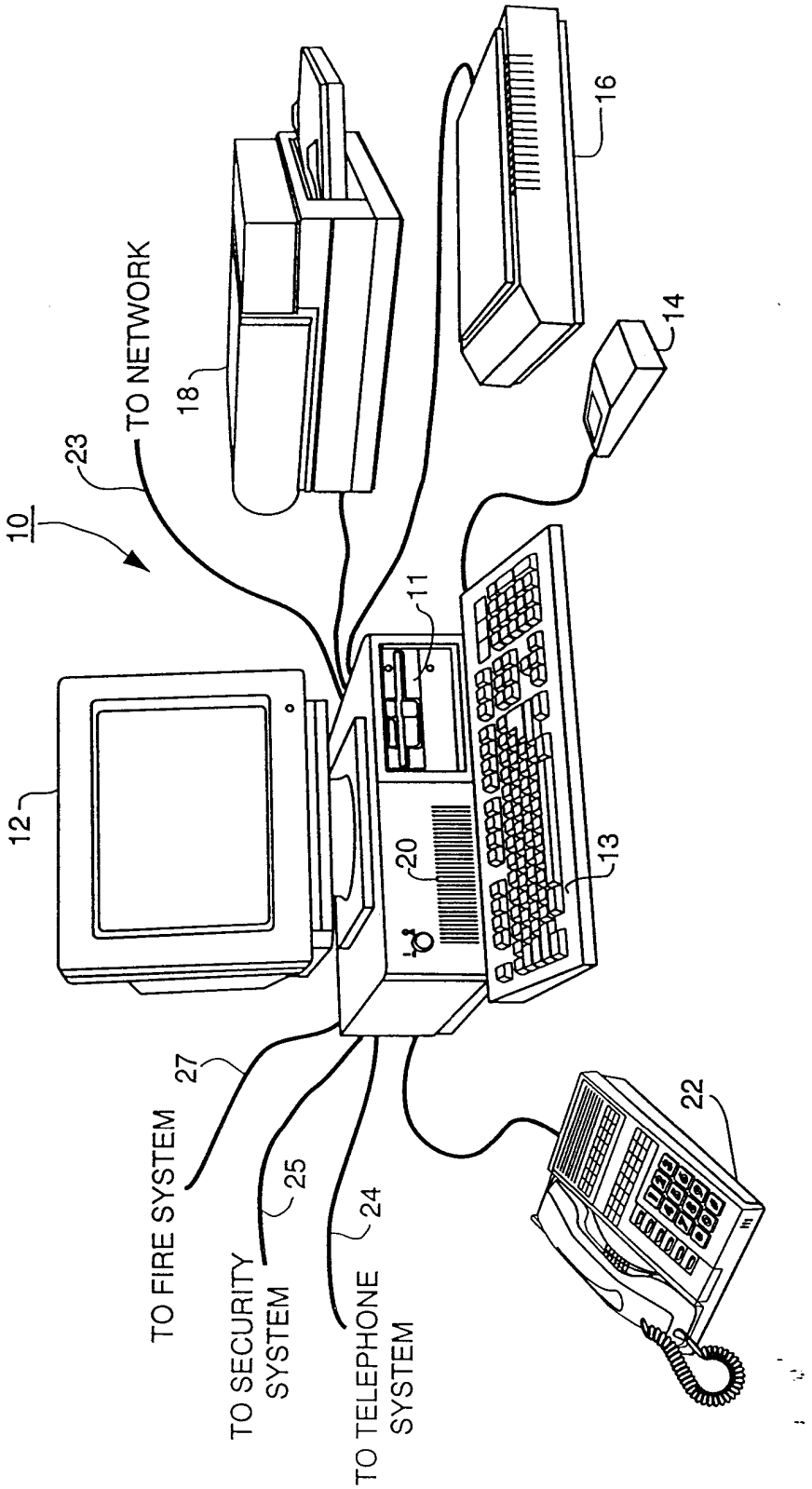
10
12
11
13
14
16
18
20
22
23
TO NETWORK
27
TO FIRE SYSTEM
25
TO SECURITY SYSTEM
24
TO TELEPHONE SYSTEM

FIG.1

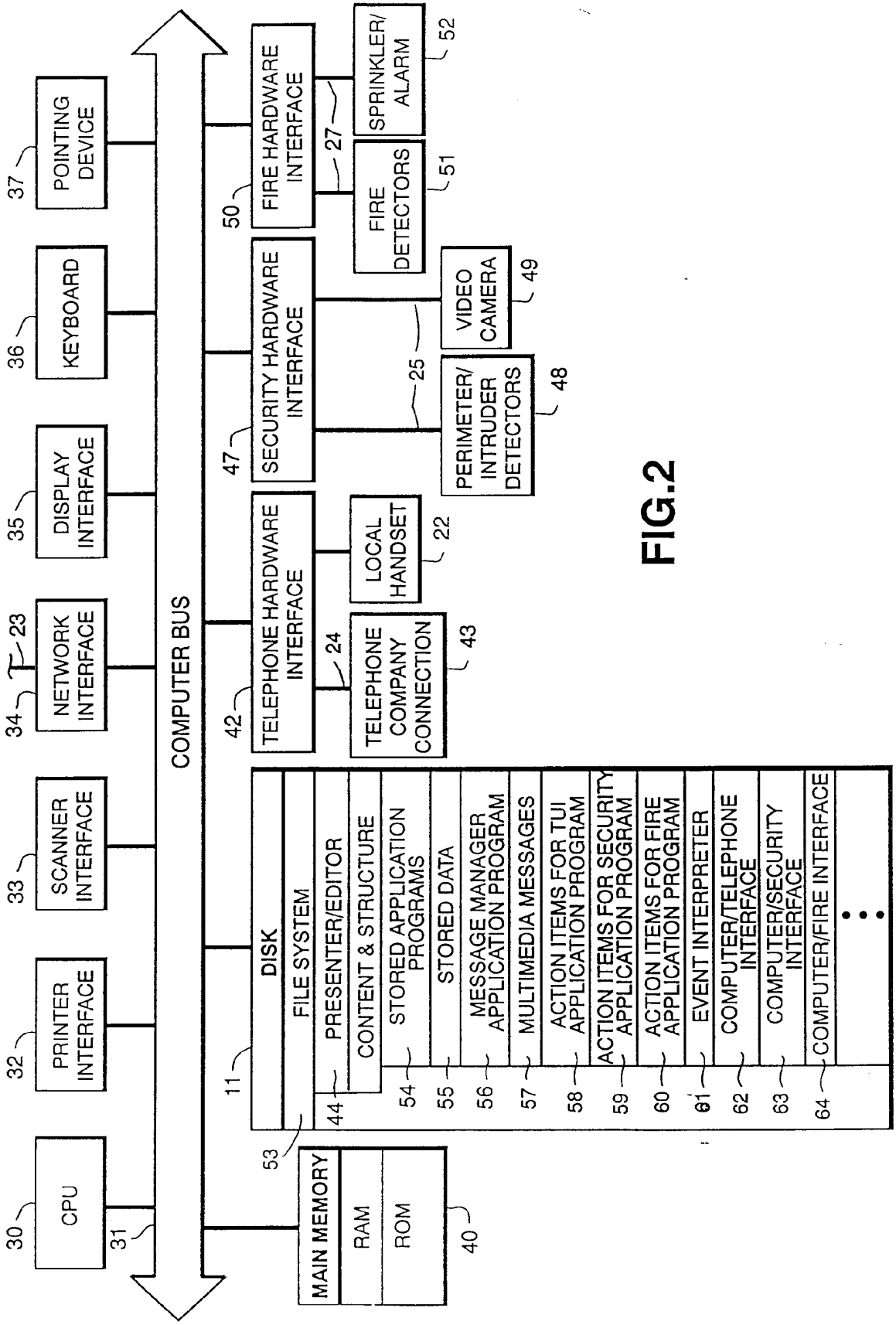
30
CPU
31
COMPUTER BUS
32
PRINTER INTERFACE
33
SCANNER INTERFACE
34
NETWORK INTERFACE
23
35
DISPLAY INTERFACE
36
KEYBOARD
37
POINTING DEVICE
MAIN MEMORY
RAM
ROM
40
53
11
DISK
FILE SYSTEM
PRESENTER/EDITOR
CONTENT & STRUCTURE
44
STORED APPLICATION PROGRAMS
54
STORED DATA
55
MESSAGE MANAGER APPLICATION PROGRAM
56
MULTIMEDIA MESSAGES
57
ACTION ITEMS FOR TUI APPLICATION PROGRAM
58
ACTION ITEMS FOR SECURITY APPLICATION PROGRAM
59
ACTION ITEMS FOR FIRE APPLICATION PROGRAM
60
EVENT INTERPRETER
61
COMPUTER/TELEPHONE INTERFACE
62
COMPUTER/SECURITY INTERFACE
63
COMPUTER/FIRE INTERFACE
64
42
TELEPHONE HARDWARE INTERFACE
24
TELEPHONE COMPANY CONNECTION
43
LOCAL HANDSET
22
47
SECURITY HARDWARE INTERFACE
25
PERIMETER/ INTRUDER DETECTORS
48
VIDEO CAMERA
49
50
FIRE HARDWARE INTERFACE
27
FIRE DETECTORS
51
SPRINKLER/ ALARM
52

FIG.2

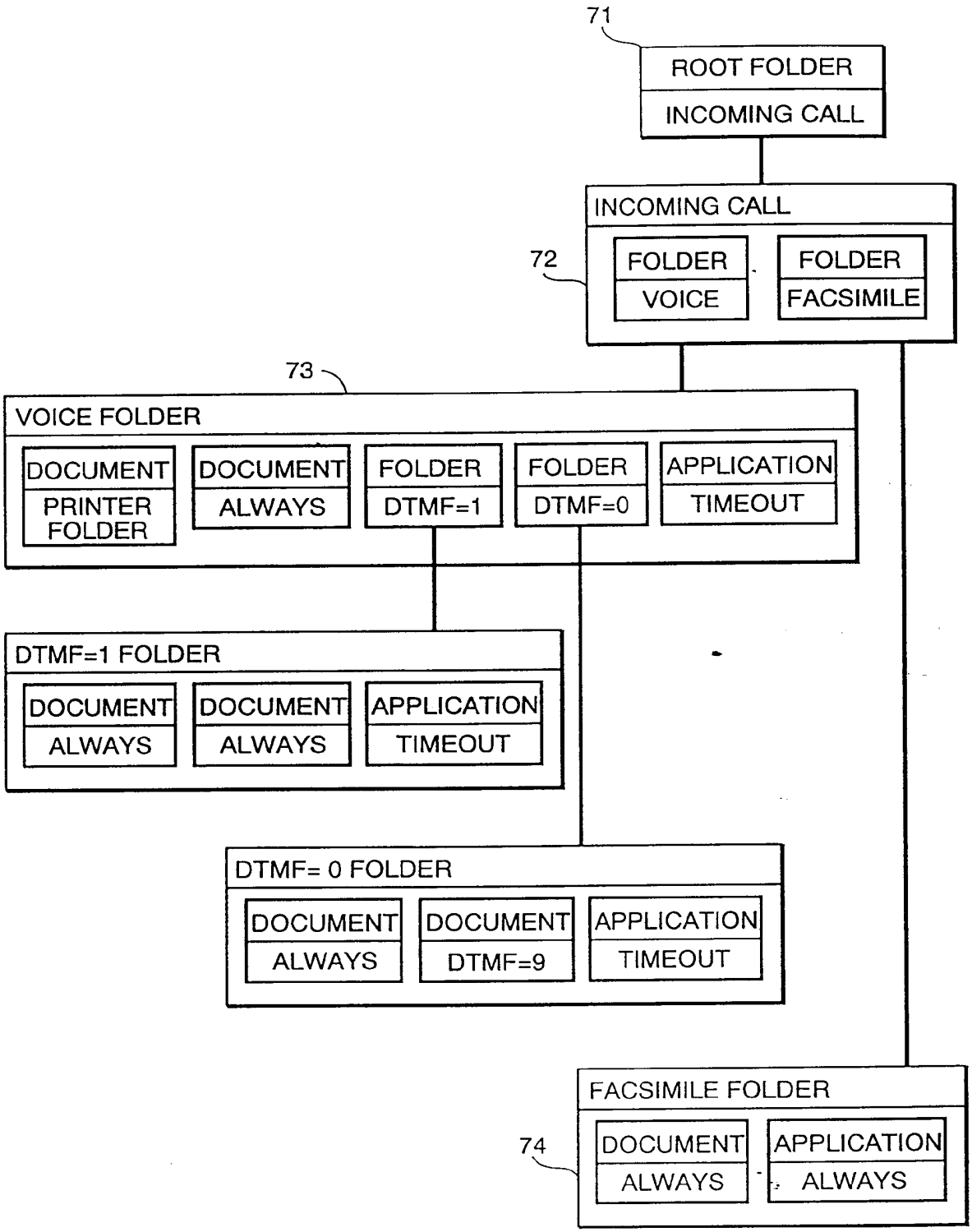
71
ROOT FOLDER
INCOMING CALL
INCOMING CALL
72
FOLDER
VOICE
FOLDER
FACSIMILE
73
VOICE FOLDER
DOCUMENT
PRINTER FOLDER
DOCUMENT
ALWAYS
FOLDER
DTMF=1
FOLDER
DTMF=0
APPLICATION
TIMEOUT
DTMF=1 FOLDER
DOCUMENT
ALWAYS
DOCUMENT
ALWAYS
APPLICATION
TIMEOUT
DTMF= 0 FOLDER
DOCUMENT
ALWAYS
DOCUMENT
DTMF=9
APPLICATION
TIMEOUT
FACSIMILE FOLDER
74
DOCUMENT
ALWAYS
APPLICATION
ALWAYS

FIG.3

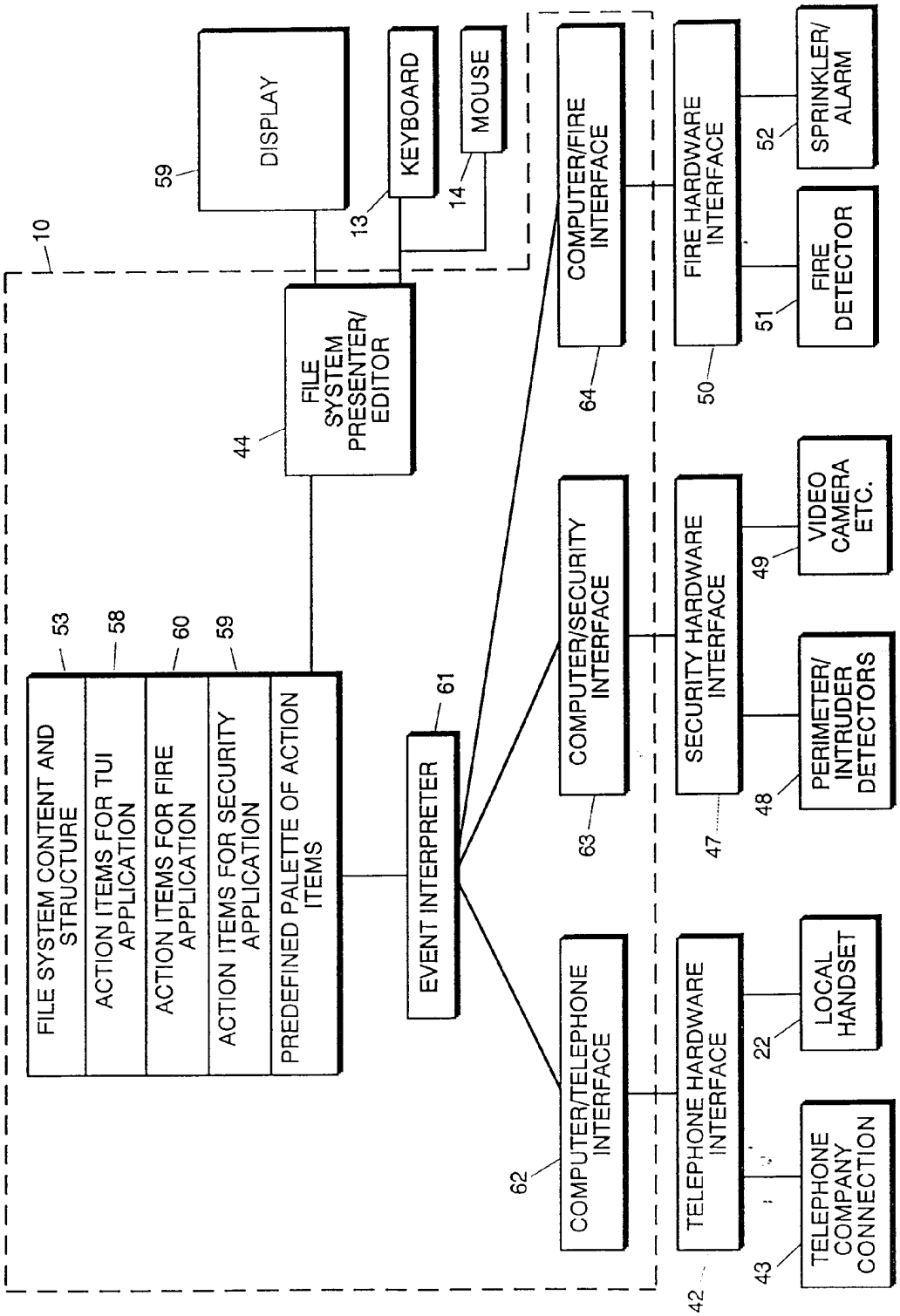
10
59
DISPLAY
13
KEYBOARD
14
MOUSE
44
FILE SYSTEM PRESENTER/ EDITOR
53
58
60
59
FILE SYSTEM CONTENT AND STRUCTURE
ACTION ITEMS FOR TUI APPLICATION
ACTION ITEMS FOR FIRE APPLICATION
ACTION ITEMS FOR SECURITY APPLICATION
PREDEFINED PALETTE OF ACTION ITEMS
61
EVENT INTERPRETER
62
COMPUTER/TELEPHONE INTERFACE
63
COMPUTER/SECURITY INTERFACE
64
COMPUTER/FIRE INTERFACE
42
TELEPHONE HARDWARE INTERFACE
22
LOCAL HANDSET
43
TELEPHONE COMPANY CONNECTION
47
SECURITY HARDWARE INTERFACE
49
VIDEO CAMERA ETC.
48
PERIMETER/ INTRUDER DETECTORS
50
FIRE HARDWARE INTERFACE
51
FIRE DETECTOR
52
SPRINKLER/ ALARM

FIG.4

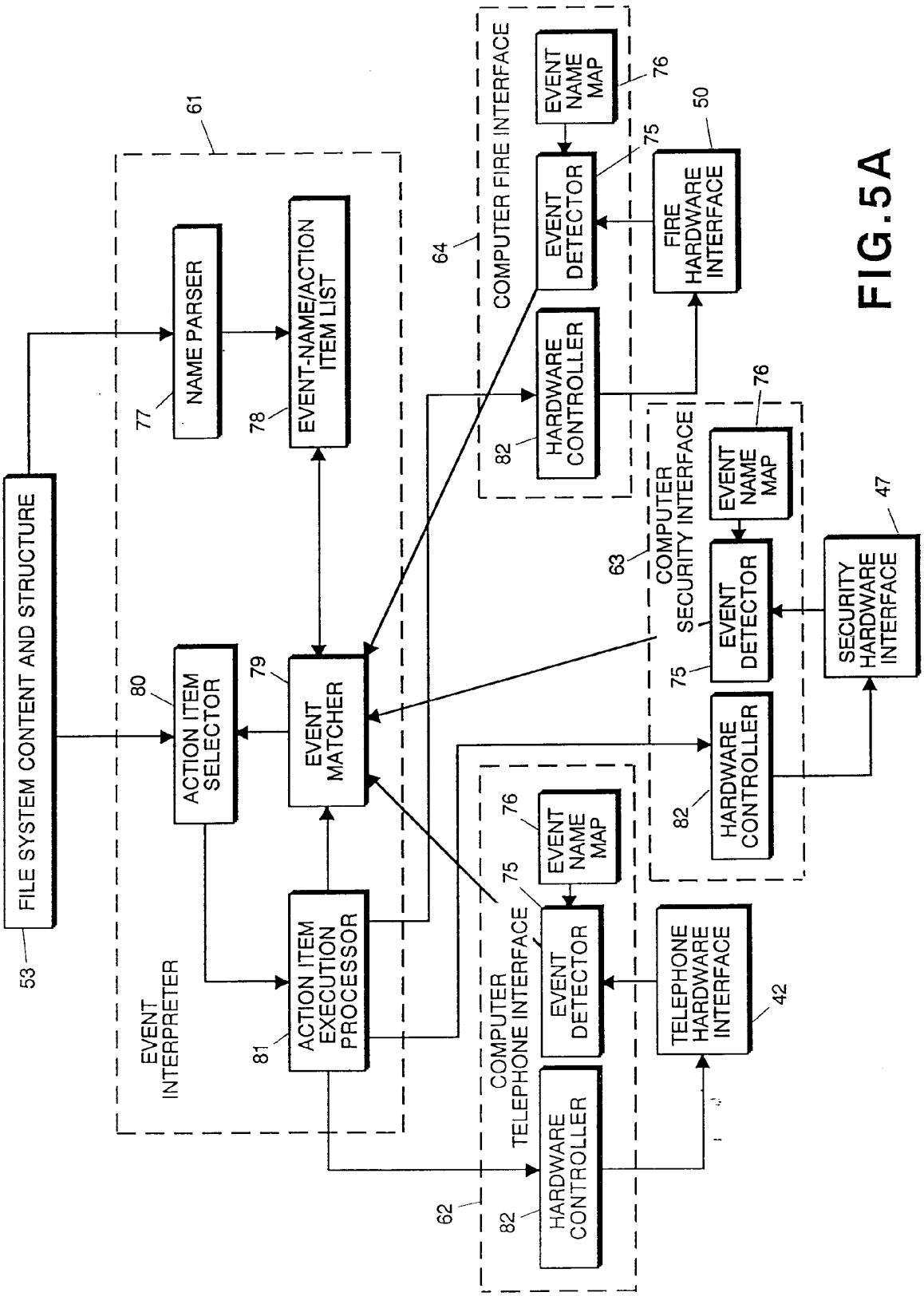
FILE SYSTEM CONTENT AND STRUCTURE
53
EVENT INTERPRETER
61
NAME PARSER
77
EVENT-NAME/ACTION ITEM LIST
78
ACTION ITEM SELECTOR
80
EVENT MATCHER
79
ACTION ITEM EXECUTION PROCESSOR
81
COMPUTER FIRE INTERFACE
64
EVENT NAME MAP
76
EVENT DETECTOR
75
FIRE HARDWARE INTERFACE
50
HARDWARE CONTROLLER
82
COMPUTER SECURITY INTERFACE
63
EVENT NAME MAP
76
EVENT DETECTOR
75
SECURITY HARDWARE INTERFACE
47
HARDWARE CONTROLLER
82
COMPUTER TELEPHONE INTERFACE
62
EVENT NAME MAP
76
EVENT DETECTOR
75
TELEPHONE HARDWARE INTERFACE
42
HARDWARE CONTROLLER
82

FIG.5A

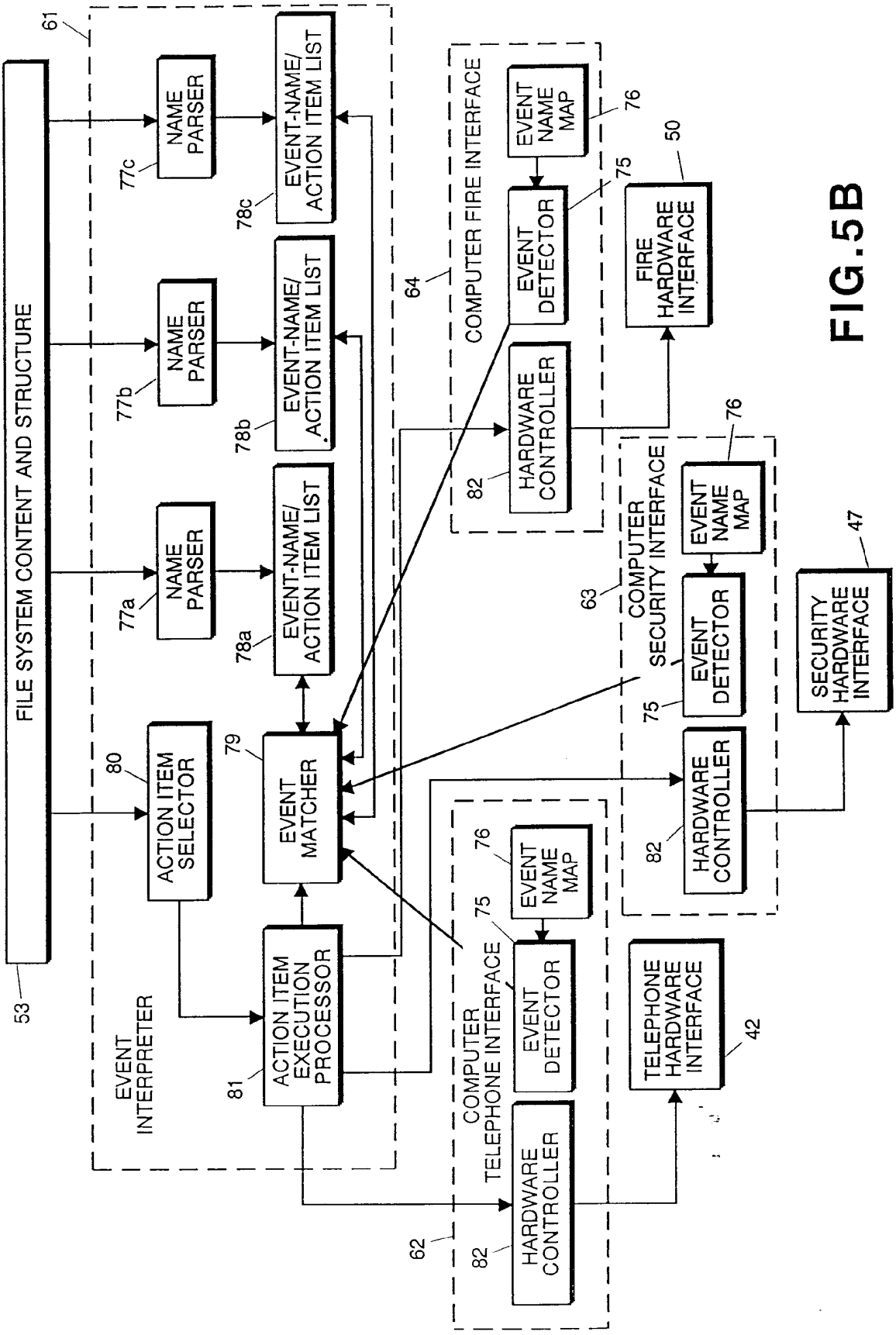
FILE SYSTEM CONTENT AND STRUCTURE
53
61
EVENT INTERPRETER
NAME PARSER
77a
77b
77c
EVENT-NAME/ ACTION ITEM LIST
78a
78b
78c
ACTION ITEM SELECTOR
80
EVENT MATCHER
79
ACTION ITEM EXECUTION PROCESSOR
81
COMPUTER TELEPHONE INTERFACE
62
HARDWARE CONTROLLER
82
EVENT DETECTOR
75
EVENT NAME MAP
76
TELEPHONE HARDWARE INTERFACE
42
COMPUTER FIRE INTERFACE
64
FIRE HARDWARE INTERFACE
50
COMPUTER SECURITY INTERFACE
63
SECURITY HARDWARE INTERFACE
47

FIG.5B

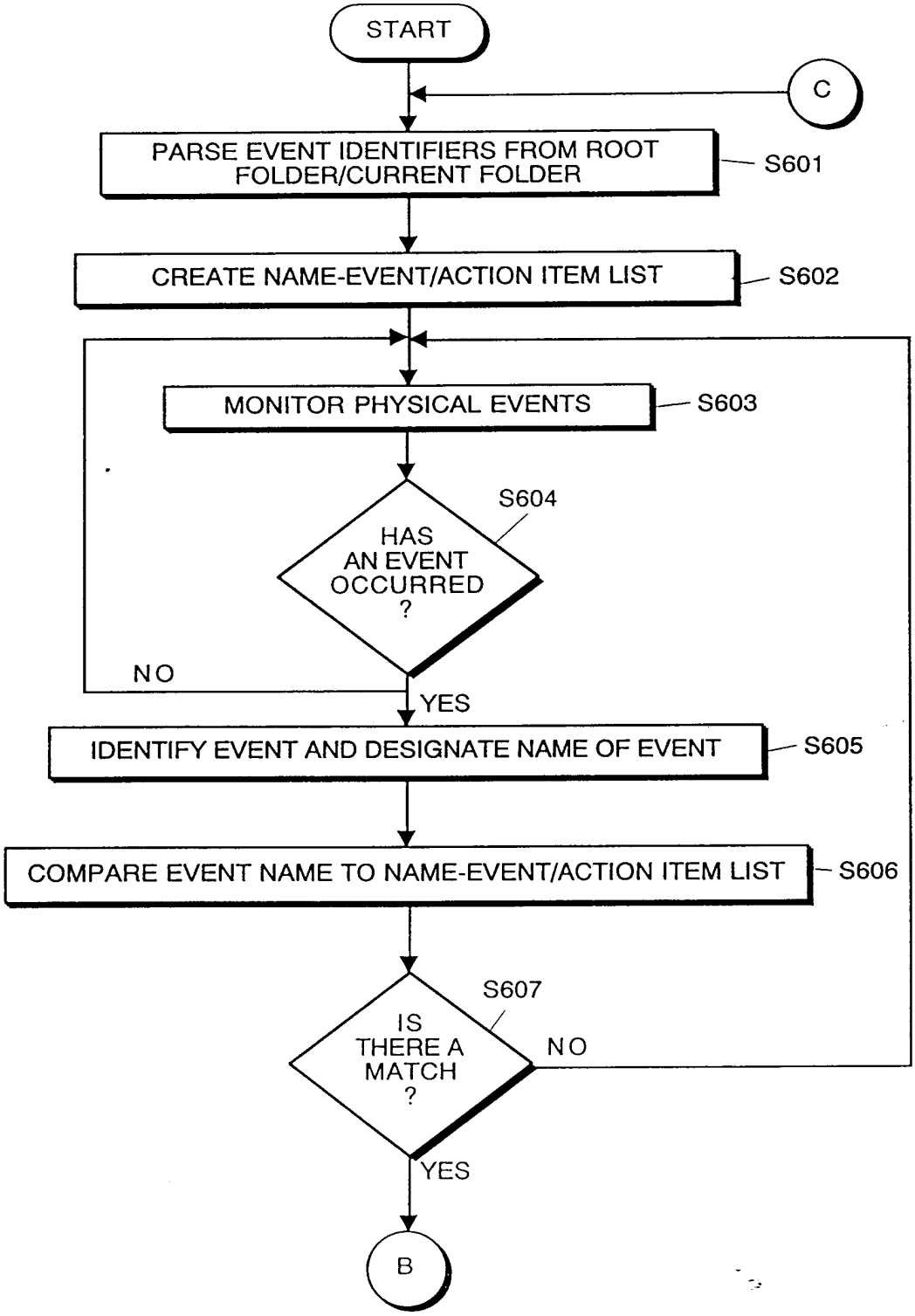
START
C
PARSE EVENT IDENTIFIERS FROM ROOT FOLDER/CURRENT FOLDER
S601
CREATE NAME-EVENT/ACTION ITEM LIST
S602
MONITOR PHYSICAL EVENTS
S603
S604
HAS AN EVENT OCCURRED ?
NO
YES
IDENTIFY EVENT AND DESIGNATE NAME OF EVENT
S605
COMPARE EVENT NAME TO NAME-EVENT/ACTION ITEM LIST
S606
S607
IS THERE A MATCH ?
NO
YES
B

FIG.6A

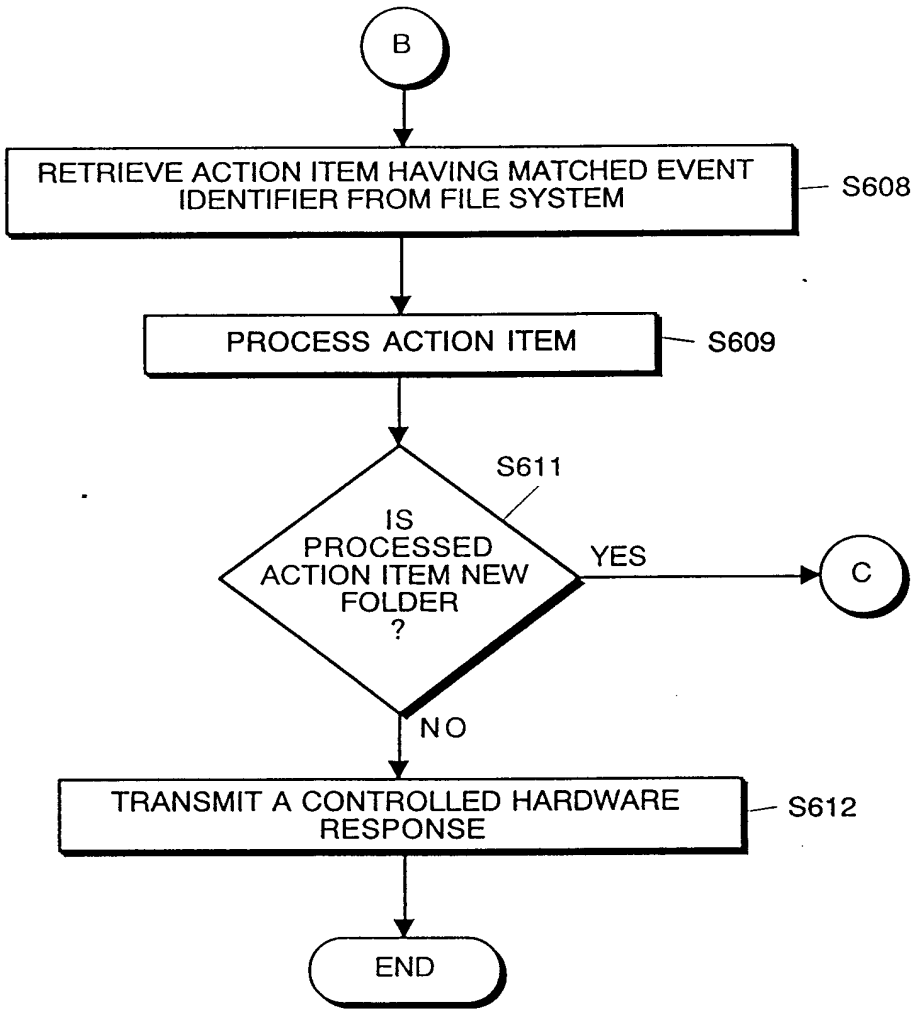
B
RETRIEVE ACTION ITEM HAVING MATCHED EVENT IDENTIFIER FROM FILE SYSTEM
S608
PROCESS ACTION ITEM
S609
S611
IS PROCESSED ACTION ITEM NEW FOLDER ?
YES
C
NO
TRANSMIT A CONTROLLED HARDWARE RESPONSE
S612
END

FIG.6B

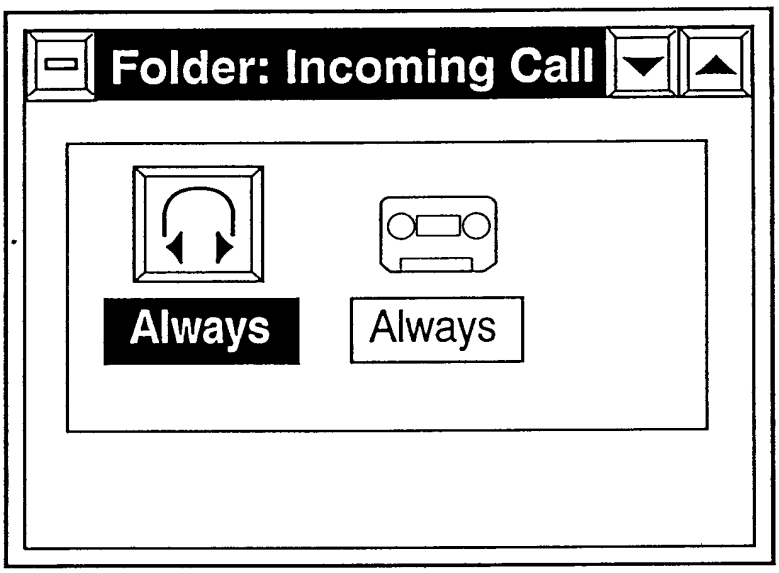

FIG.6C

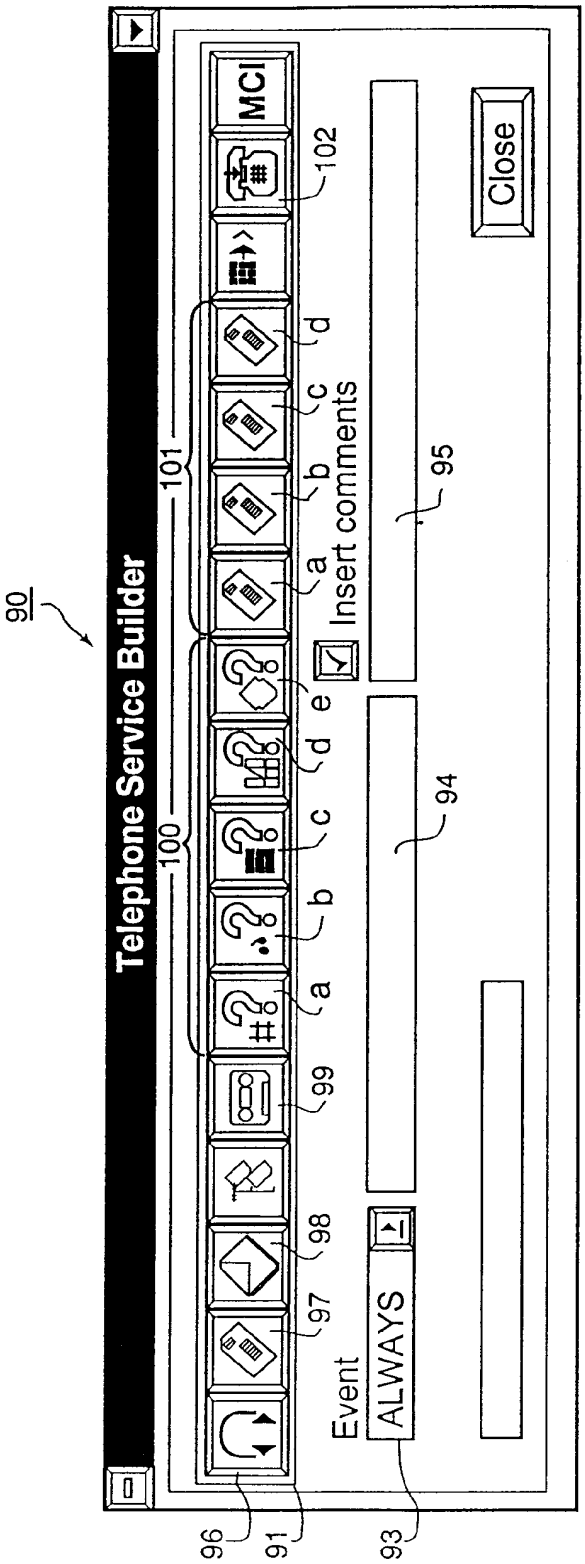
90
Telephone Service Builder
MCI
Close
Insert comments
Event
ALWAYS
100
101
102
95
94
99
98
97
96
91
93
a
b
c
d
e

FIG.7

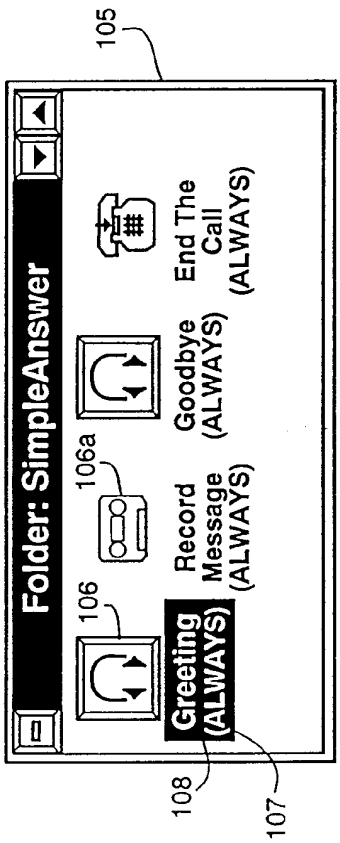
105
Folder: SimpleAnswer
106
106a
Greeting (ALWAYS)
Record Message (ALWAYS)
Goodbye (ALWAYS)
End The Call (ALWAYS)
108
107

FIG.7A

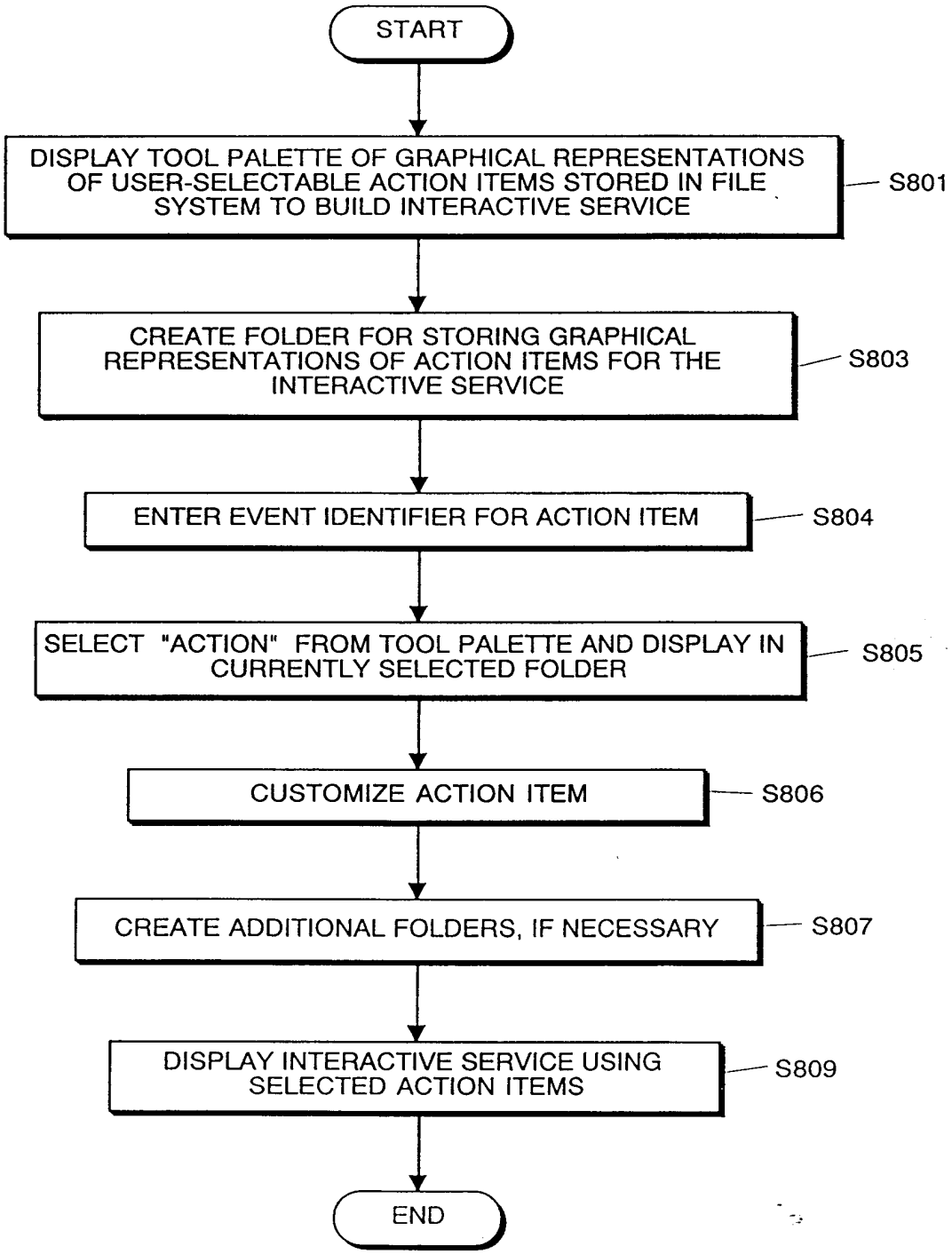
START
DISPLAY TOOL PALETTE OF GRAPHICAL REPRESENTATIONS OF USER-SELECTABLE ACTION ITEMS STORED IN FILE SYSTEM TO BUILD INTERACTIVE SERVICE
S801
CREATE FOLDER FOR STORING GRAPHICAL REPRESENTATIONS OF ACTION ITEMS FOR THE INTERACTIVE SERVICE
S803
ENTER EVENT IDENTIFIER FOR ACTION ITEM
S804
SELECT "ACTION" FROM TOOL PALETTE AND DISPLAY IN CURRENTLY SELECTED FOLDER
S805
CUSTOMIZE ACTION ITEM
S806
CREATE ADDITIONAL FOLDERS, IF NECESSARY
S807
DISPLAY INTERACTIVE SERVICE USING SELECTED ACTION ITEMS
S809
END

FIG.8

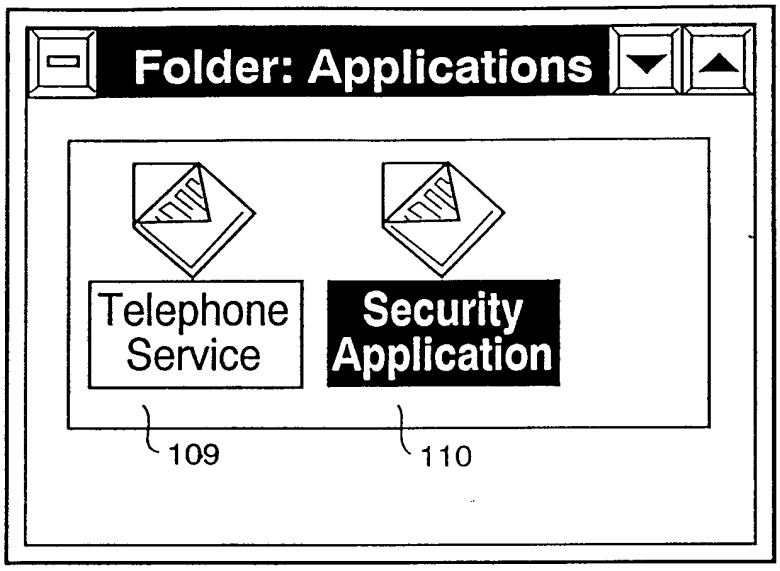
Folder: Applications
Telephone Service
Security Application
109
110

FIG.9

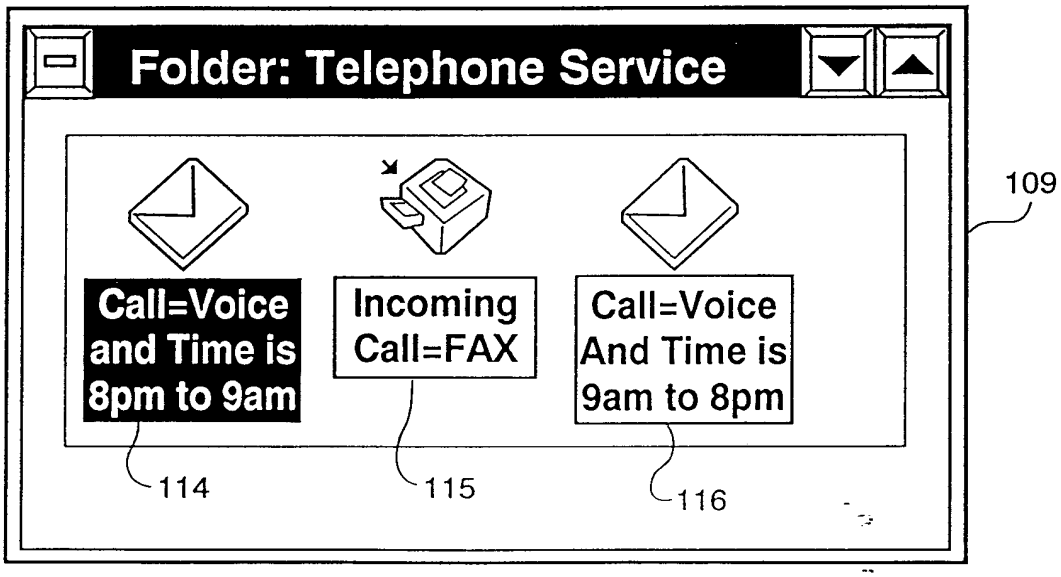
Folder: Telephone Service
Call=Voice and Time is 8pm to 9am
Incoming Call=FAX
Call=Voice And Time is 9am to 8pm
109
114
115
116

FIG.10

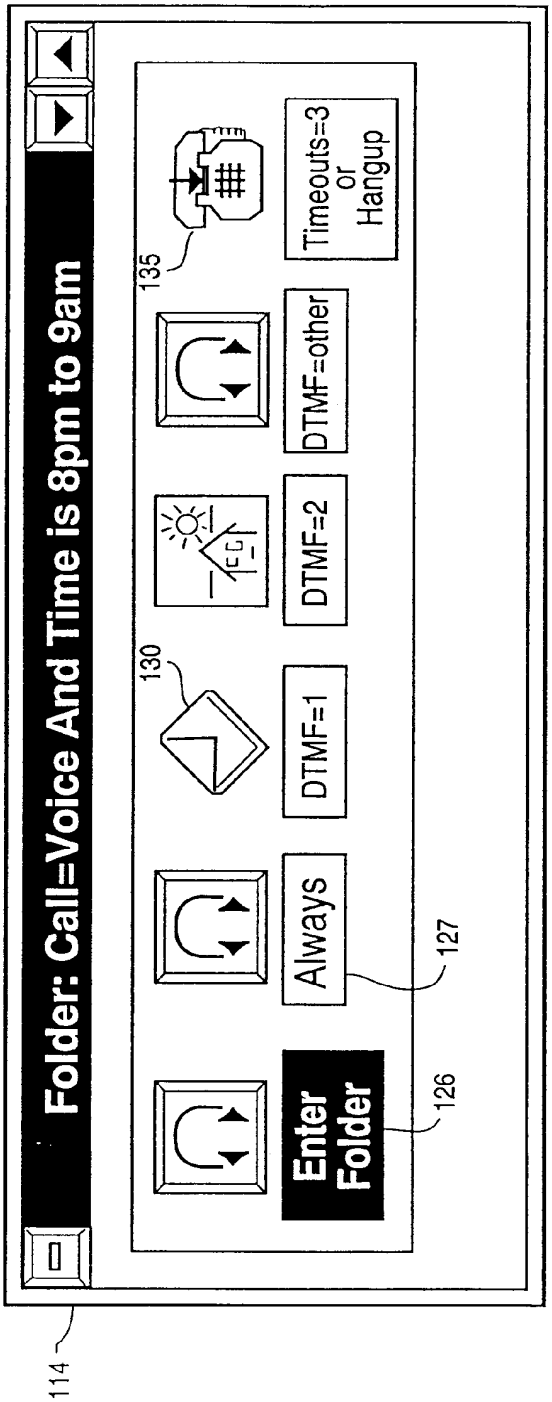

FIG.11A

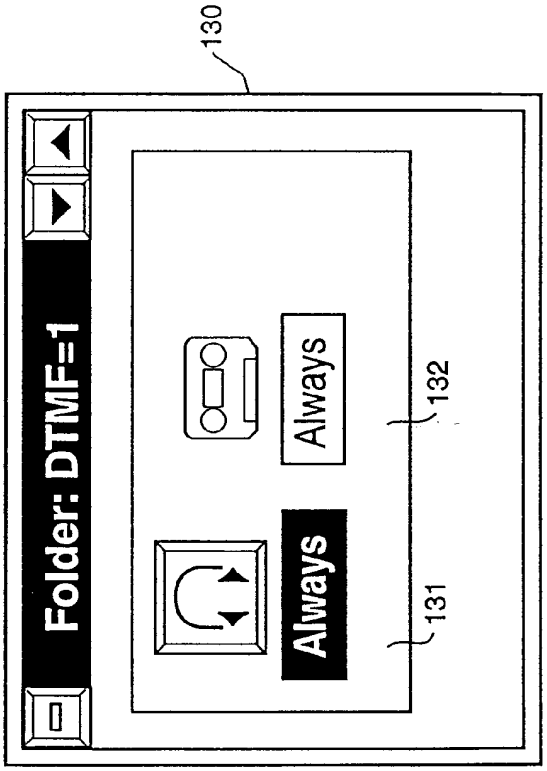

FIG.11B

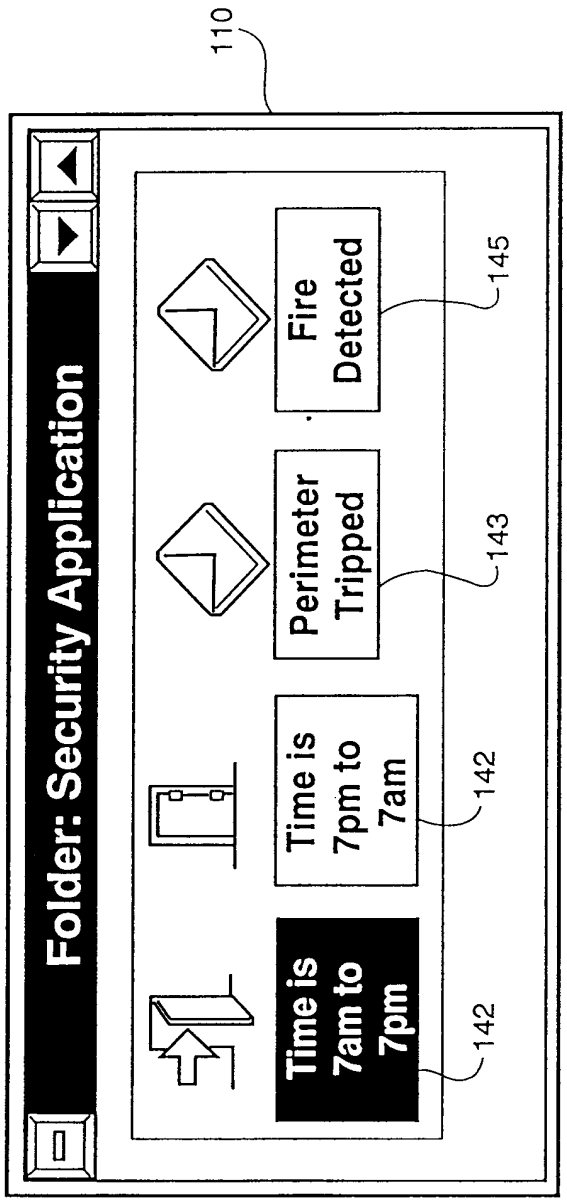

FIG.12A

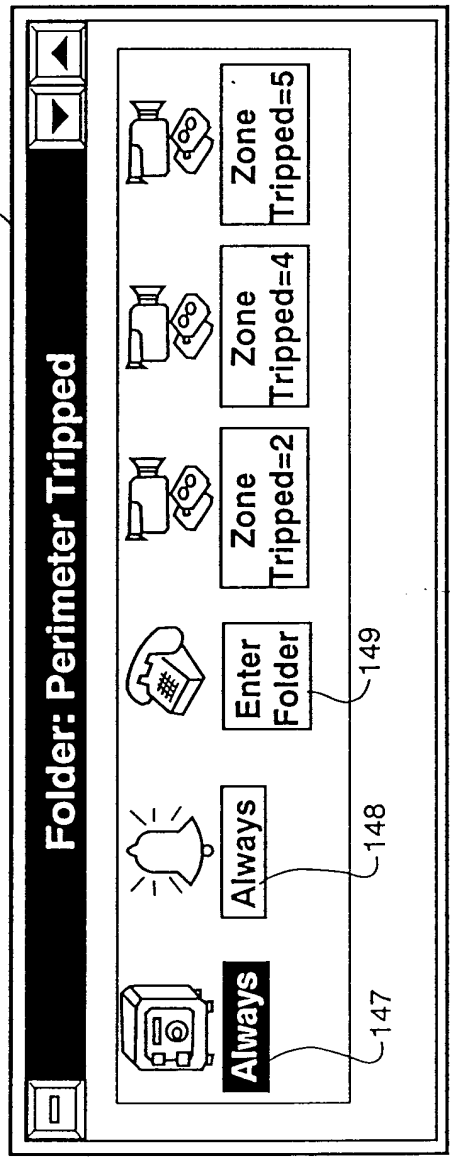

FIG.12B

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 94 30 9029

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.6)
X	PATENT ABSTRACTS OF JAPAN vol. 16, no. 355 (P-1394) 30 July 1992 & JP-A-04 107 629 (NEC CORPORATION) 9 April 1992 * abstract * ---	1,5	G06F9/46
X	PATENT ABSTRACTS OF JAPAN vol. 16, no. 365 (P-1397) 6 August 1992 & JP-A-04 112 258 (NEC CORPORATION) 14 April 1992 * abstract * ---	1	
X	EP-A-0 513 553 (COMMODORE ELECTRONICS LIMITED) * column 3, line 20 - column 5, line 27 * * column 12, line 16 - column 13, line 54 * * column 15, line 32 - column 16, line 56; figures 4,5,8 * ---	27-32	
A	EP-A-0 451 963 (IBM CORPORATION) * column 3, line 4 - line 54 * * column 6, line 8 - column 8, line 2 * -----	9,17,33, 34	TECHNICAL FIELDS SEARCHED (Int.Cl.6) G06F

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	5 April 1995	McDonagh, F

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)